CircusTent Memsys Paper Reviews/Revisions

----------------------- REVIEW 1 ---------------------

SUBMISSION: 15

TITLE: CircusTent: A Benchmark Suite for Atomic Memory Operations

AUTHORS: Brody Williams, John Leidel, Xi Wang, David Donofrio and Yong Chen

----------- Overall evaluation -----------

SCORE: -1 (weak reject)

----- TEXT:

Overall, the release of the open source atomic evaluation benchmark set is a good contribution.

The paper would be strengthened by citing how much atomics are used in parallel applications and by reporting the amount of time spent in atomics in realistic parallel applications, especially HPC. This would help the reader understand the impact of an X% improvement in latency of an atomic on a real application.

BW: Not a bad thought. We did include some discussion of the percentage of atomics instructions included in the GAPBS (see paragraph below). I assume the reviewer either did not see this or found it insufficient. If needed, I can try to find some reference that more specifically discusses the cost in terms of time and cite it.

“In our previous work, we examined the GAP Benchmark Suite [4],

which is designed to replicate the memory access patterns of graph

workloads in a shared memory environment, in order to quantify

the proportion of atomic operations. Therein, we found that, across

all six included kernels, an average of 17.46% of the total instructions

were RMW atomic operation instructions [24]. “

The overview of transactional memory was interesting. However, if transactional memory is orthogonal to what you are doing why devote so much space to it?

BW: Fair point. I will be more mindful in the future, but I don’t know that the section necessarily needs to be removed/shortened now since it is already accepted.

The evaluation only studied physically shared memory -- why not distributed memory since that's what the benchmarks are also supposed to handle?

BW: We only explored the OpenMP backend in the interest of space and to focus more on the introduction of the suite/kernels themselves. Other backend additions/evaluations could be added for a journal extension.

"future compilers may be able to further optimize performance by replac- ing atomic Add instructions with CAS analogs when values are expected to be modified infrequently." -- how would they tell that this is the case. Also, This finding seems to point to a deficiency in the use of the this particular OpenMP formulation of the random access benchmark. Is there another random access benchmark in the suite that doesn't write back to the same location?

BW: I would assume, if the programmer is familiar with the codebase, they could specify such an option. Admittedly, I am otherwise unsure. Do we need to address this?

There is not currently another random-access kernel implementation. We could add to/revise the existing one in the future.

Some of the discussion doesn't identify a root cause for the anomaly, nor does it suggest a different benchmark or methodology to find a root cause.

BW: I think this only occurs in two places. Both are in the “new insights” subsection. The first is with the I7-4980HQ. The second is in regard to the fact that the OS and compiler may also play a role (last paragraph). Both are acknowledged. Given the number of platforms, is this a serious transgression that needs to be fixed?

----------------------- REVIEW 2 ---------------------

SUBMISSION: 15

TITLE: CircusTent: A Benchmark Suite for Atomic Memory Operations

AUTHORS: Brody Williams, John Leidel, Xi Wang, David Donofrio and Yong Chen

----------- Overall evaluation -----------

SCORE: 2 (accept)

----- TEXT:

Good topic. The charts (Fig 2-9) are almost unreadable, probably because they try to display too many configurations (?) This needs to be fixed.

BW: Agreed. The figures are difficult to decipher due to the number test systems and the range of cores on the x-axis (2-72).

I’m not sure what the best approach to remedying this situation would be. I tried switch to log-scale graphs, but that did not help. We could enlarge the symbols and/or use common colors for similar architectures as reviewer 3 suggests, but it may still be difficult to read due to the number of systems.

Xi suggested we could cut some of the systems/results out for a future journal extension. That might be possible, but I suppose we would still face the same issue in the future. The KNL system would be the best system to remove if we go that route. However, it would also entail removing some of the more interesting results/analysis discussions.

I have some “zoomed” versions of the figures already created as well (overleaf->extended\_figures), but I don’t know if including multiple versions of the same data is advisable/acceptable.

Alternatively, we could altogether hide some of the lines in each figure. Perhaps include only systems discussed in the text for each kernel. Again, I don’t know how well that would play if someone is interested in the performance of a particular system across benchmarks.

What do you guys suggest in this regard?

----------------------- REVIEW 3 ---------------------

SUBMISSION: 15

TITLE: CircusTent: A Benchmark Suite for Atomic Memory Operations

AUTHORS: Brody Williams, John Leidel, Xi Wang, David Donofrio and Yong Chen

----------- Overall evaluation -----------

SCORE: 2 (accept)

----- TEXT:

Paper Summary:

This paper presents CircusTent, a new benchmark suite for evaluating atomic

operations with a variety of memory access patterns in both shared and

distributed memory environments. The paper also presents detailed evaluation

of the performance of atomic operations on a variety of modern computing

platforms using their benchmark.

Strengths:

The presented benchmark suite is well thought out and supports performance

characterization of multiple atomic operations with different access patterns

across a variety of backend architectures.

The paper presents a comprehensive evaluation of atomic operations on a large

set of computing architectures. The results and analysis provide a useful

comparison of the performance and scalability of atomic operations on these

architectures.

Weaknesses:

The figures in Section 4.3 are extremely difficult to read due to the large

number of lines in each figure. Please consider making the markers larger,

using common colors for architectures with similar features, and/or limiting

the number of results you present in each figure.

BW: See discussion of figures above.

Although the earlier sections note that the benchmark supports multiple

backends, the evaluation only considers the OpenMP backend. I think you should

make it clear in the intro that this work only evaluates the OpenMP backend.

BW: In the last paragraph of the abstract we state:

“We then demonstrate the capabilities of CircusTent through an evaluation of fourteen different platforms using our OpenMP benchmark implementation.”

Similarly, the end of introduction has:

“Section 4 presents an evaluation of the CircusTent benchmark suite conducted using a variety of different architectures and the OpenMP programming model.”

For the beginning of the benchmark section, I have added the bolded text below:

“In order to verify and demonstrate the capabilities of the CircusTent benchmark suite, we conducted an evaluation of a diverse set of platforms with respect to atomic memory operations **using our OpenMP backend implementation**.”

Should that be sufficient?

One other note:

The line that starts:

'Although the latency of this MCDRAM is a significant improvement over that of

main memory, ...' (p. 12, col. 1, paragraph 2) is not correct.

The latency of MCDRAM is about the same, or even slightly worse, than DDR on

the KNL. (e.g., see 'Exploring the Performance Benefit of Hybrid Memory System

on HPC Environments' by Peng et al. (IPDPSW 2017)). The advantage of MCDRAM is

that it is able to sustain much higher bandwidth than DDR.

BW: This was an assumption on my part. That is an interesting characteristic. Kudos to the reviewer, the paper is a great reference for the behavior.

Revised:

”Although the latency of this MCDRAM is a significant improvement over that of main memory, it is also considerably poorer than that of a true L3 cache. As such, any requests that would be serviced by the L3 cache or main memory on other systems incur a significant penalty for the Xeon Phi 7250 platform.”

->

“Although one might expect this MCDRAM to improve performance in comparison to main memory, the opposite effect occurs. Here, the increased latency of the MCDRAM combined with the irregular memory access patterns exhibited by the CircusTent kernels diminish the platform's performance~\cite{peng2017exploring}. As such, any requests that would be serviced by the L3 cache, or even main memory, on other systems often incur a significant penalty for the Xeon Phi 7250 platform.

Review Summary:

Performance of atomics is an interesting topic, but I personally have not seen

much work in this area. I think the benchmarks and tools presented in this

paper will be of interest to people at MemSys and have potential to be useful

to the larger performance community.